AIAA 2013-4699
Guidance, Navigation, and Control and Co-located Conferences
August 19-22, 2013, Boston, MA
AIAA Guidance, Navigation, and Control (GNC) Conference

Performance of a Monocular Vision-aided Inertial Navigation System for a Small UAV

Daniel Magree*, Eric N. Johnson †

The use of optical sensors for navigation on aircraft has receive much attention recently. Optical sensors provide a wealth of information about the environment and are standard payloads for many unmanned aerial vehicles (UAVs). Simultaneous localization and mapping (SLAM) algorithms using optical sensors have become computationally feasible in real time in the last ten years. However, implementations of visual SLAM navigation systems on aerial vehicles are still new and consequently are often limited to restrictive environments or idealized conditions. One example of a flight condition which can dramatically affect navigation performance is altitude. This paper seeks to examine the performance of monocular extended Kalman filter based SLAM (EKF-SLAM) navigation over a large altitude change. Simulation data is collected which illustrates the behavior of the navigation system over the altitude range. Navigation and control system parameters values are specified which improve vehicle performance across the flight conditions. Additionally, a detailed presentation of the monocular EKF-SLAM navigation system is given. Flight test results are presented on a quadrotor.

I. Introduction

Today we are seeing a rapid expansion of the use of unmanned aerial vehicles taking on complex tasks. UAVs are being used to explore cluttered, unmapped, environments, inspect infrastructure, and operate in urban and natural canyons. All of these tasks require accurate and precise navigation system for safe operation in close quarters. The most common sensor used for bounding the position estimate of UAVs today is the global positioning system (GPS). GPS relies on an external infrastructure of satellites to triangulate the position of the receiver. The use of GPS requires the reception of the data signal from a sufficient number of satellites to calculate a reliable position. Today UAVs are operating in cluttered environments where GPS signals are weak or obstructed, or in hostile environments where the GPS signal could be jammed. Alternative sensors and algorithms for navigation are necessary to effectively operate in these areas.

Vision-based navigation is an attractive alternative due to the wealth of information provided by a camera. Cameras are often standard payloads on UAVs, and are both low cost and light weight. In the last 10 years, computational power has increased to the point of allowing the necessary image processing to be performed in real time. However, implementations of vision-based navigation techniques on aerospace vehicles are still new. Little work has been done to explore their behavior over the range of environments that a UAV is likely to encounter. This paper seeks to begin this discussion by characterizing the behavior of a vision-based navigation system over a large altitude change, and identify methods to improve performance for the entire flight envelope.

The navigation algorithm presented in this paper falls into the category of simultaneous localization and mapping (SLAM) systems. Monocular SLAM navigation was first successfully implemented by Davidson et al. in 2003.[1] This system used the camera as the only sensor, and therefore could not independently measure the scale of the map. Since this time, several paradigms have emerged: Keyframe-based SLAM,[2] EKF-SLAM,[3] and particle filter SLAM.[4] Additional extensions to these paradigms improve robustness and

*Graduate Research Assistant, Georgia Institute of Technology, dmagree@gatech.edu.
†Associate Professor, Georgia Institute of Technology, eric.johnson@ae.gatech.edu.

scalability, such as inverse depth parametrization presented in Civera et al.,[5] and relative bundle adjustment presented in Sibley.[6] The results in these papers present open-loop data from cameras carried or driven by an operator, in particular, the SLAM provided state estimates are not used for closed-loop control on vehicles with fast dynamics.

More recently, implementations of visual SLAM navigation for aircraft have been developed. These implementations have focused on navigation accuracy during extended flight times and distances. In Chawdhary et al. a monocular vision-aided navigation system is presented, capable of maintaining vehicle stability indefinitely.[7] Flight test results of the algorithm are presented for a 66 kg helicopter as well as a 1 kg ducted fan vehicle. A flight of over 16 minutes was performed, and the vehicle performed an autonomous landing using the vision-based navigation system. In Weiss et al. a monocular slam implementation and LQR controller is described,.[8] The SLAM algorithm used is based on PTAM algorithm,[2] a keyframe based method using periodic global map optimization and localization from the resulting map. The results presented show the vehicle is capable of waypoint following flight and accurate position hold. These results are expanded in Refs. 9 and 10. While the extended distances covered and the waypoint following flight are significant accomplishments, the flight trajectories followed do not change the operating conditions of the vision system in ways that will affect performance.

Other works exploring challenging navigation scenarios include the work by Scherer et al.,[11] in which navigation in the presence of intermittent GPS signals was explored. This work uses a stereo camera sensor along with a laser range finder and intermittent GPS to map riverines, in the presence of obstacles such as trees, bridges, and embankments. The results are presented from a surrogate data collection system mounted on a small boat, and show an impressive ability to navigate in adverse conditions. This work did not specifically address how changing conditions affected the quality of the vision-based navigation solution, which is the focus of this paper.

This work builds on previous papers developing the vision-aided inertial navigation system. Wu et al. developed a method for fusing feature information from a monocular vision sensor in an extended Kalman filter framework.[12,13] The approach in those papers relied on tracking features whose locations were estimated when GPS was active. When GPS was inactive, they demonstrated that this method ensures bounded hover of a rotorcraft UAV through flight test results. This work was continued in Chawdhary et al. in which a fully independent vision-aided inertial navigation system was presented and flight tested.[7]

In this paper, a flight profile is examined which is designed to challenge the vision-based navigation system developed in Chawdhary et al. Over a large change in altitude, the uncertainty of the vision estimate changes significantly, and this change is shown to cause failure of the closed loop system if not dealt with appropriately. In the next section, an overview of the vision-based navigation system is presented. Next, simulation results over the large altitude change are presented, and performance of the vehicle over a range of navigation settings is examined. Finally, flight test results of the system are presented which validate the simulation study. The flight tests are performed on the GTQ3, a quadrotor vehicle weighing 1.5 kg and performing all processing onboard in real time. Figure 1 shows a picture of the GTQ3.

II. SLAM Navigation System

This section presents an overview of the visual SLAM navigation system. The foundation of this system has been described in detail in Ref. 7, and this paper describes several improvements in the original design. Most significant is the use of a feature descriptor during point matching.

The visual SLAM navigation system uses an extended Kalman filter to estimate the state of the vehicle and the location of image features in the environment. Images are captured and features are extracted from the images. The features from the first image are used to initialize a database of three-dimensional point locations. When each subsequent image is captured, features are extracted and matched to the points stored in the database, the error between their predicted and measured location is used to update their location

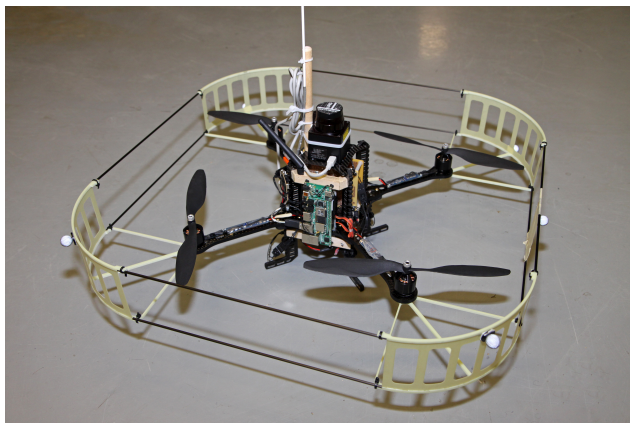


Figure 1. The GTQ3 weighs 1.5 kg and performs all guidance, navigation and control tasks onboard and in real time. The camera is mounted underneath the vehicle.

and the vehicle state via the extended Kalman filter gain and update equations. Points that are no longer visible due to vehicle motion are discarded and unmatched measurements are initialized into the database. Between image captures, the vehicle state is propagated by integrating data from the inertial measurement unit (IMU). A schematic of the algorithm is shown in Figure 2.

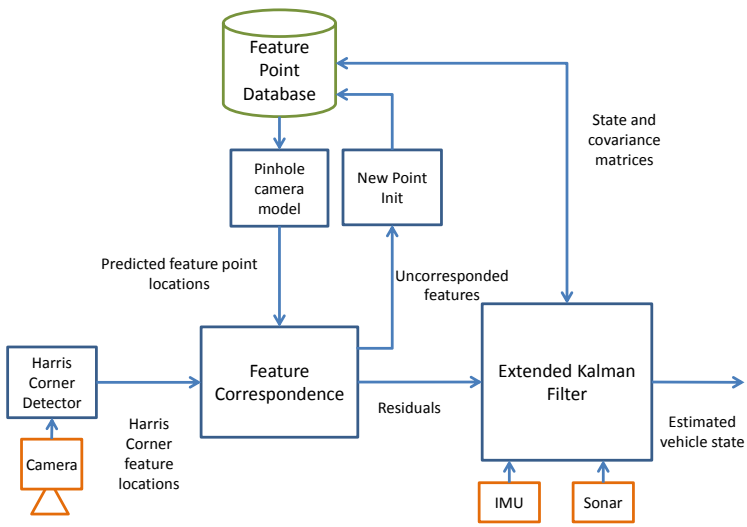


Figure 2. A block diagram describing the navigation system. Pixel location of features are corresponded to predicted locations of database points. The resulting residual is fused with inertial data and sonar altimeter data to estimate the vehicle state, feature locations and associated covariance matrices. Uncorresponded features can be used to update the database as current points leave the field of view. The resulting architecture simultaneously estimates feature point locations and vehicle states without requiring the vehicle to maintain view of particular landmarks.

A. Propagation of the estimator

The vehicle state is given below:

$$\hat{x}_a = \begin{bmatrix} \hat{p}^i & \hat{v}^i & \hat{q}^i & \hat{s}_b & \hat{\omega}_b \end{bmatrix}^{\mathrm{T}} \tag{1}$$

where p, v, q, is the vehicle position, velocity and attitude quaternion, respectively, s_b is the acceleration bias and ω_b is the gyro bias. Superscript i denotes the inertial frame and b denotes body frame and hatted variables indicate estimated quantities. The rotation matrix from body to inertial is denoted $L_{ib} = L_{bi}^{\mathrm{T}}$. The vehicle state is propagated by integrating data from the IMU. IMU sensor measurements are corrupted by noise and bias as follows:

$$s_{raw} = a + s_b + L_{bi}g + \eta_a, \tag{2}$$

$$\omega_{raw} = \omega_t + \omega_b + \eta_\omega. \tag{3}$$

where a, and ω_t are are the true acceleration and angular velocity, respectively. It is assumed that the noise is zero mean and white Gaussian, i.e. $\eta_a \sim N(0, Q_a)$ and $\eta_\omega \sim N(0, Q_\omega)$. The estimated bias is subtracted from the IMU data before propagation in the model

$$s = s_{raw} - \hat{s}_b, \tag{4}$$

$$\omega = \omega_{raw} - \hat{\omega}_b. \tag{5}$$

The vehicle dynamics are given by the following:

$$\dot{\hat{p}}^i = v^b \tag{6}$$

$$\dot{\hat{v}}^i = L_{ib}s - g \tag{7}$$

$$\dot{\hat{q}}^i = \frac{1}{2}\mathcal{Q}(\omega)\hat{q}^i \tag{8}$$

$$\dot{\hat{s}}_b = 0 \tag{9}$$

$$\dot{\hat{\omega}}_b = 0 \tag{10}$$

where s is specific force as measured by the IMU, g is the acceleration due to gravity in the inertial frame, and angular velocity ω_s is the angular velocity as measured by the IMU. The function $\mathcal{Q} : \mathbb{R}^3 \to \mathbb{R}^{4\times4}$ maps angular velocity to the quaternion derivative and is given by

$$\mathcal{Q}([p \quad q \quad r]^{\mathrm{T}}) = \begin{bmatrix} 0 & -p & -q & -r \\ p & 0 & r & -q \\ q & -r & 0 & p \\ r & q & -p & 0 \end{bmatrix} \tag{11}$$

Using the quaternion representation in the estimation algorithm causes the covariance matrix to become singular and leads to numerical difficulties. To avoid this problem a minimal representation of the vehicles attitude is used which defines the vehicle's current attitude with respect to an arbitrary reference frame, in this case the attitude in the previous time step. Since this attitude change is small, it is defined as an infinitesimal error quaternion.

$$\delta q = \begin{bmatrix} 1 & \hat{R} \end{bmatrix}^{\mathrm{T}} \tag{12}$$

such that

$$\delta q = q \otimes \hat{q}^{-1}. \tag{13}$$

The vehicle state vector estimated by the EKF is

$$\hat{x} = \begin{bmatrix} \hat{p}^i & \hat{v}^i & \hat{R} & \hat{s}_b & \hat{\omega}_b \end{bmatrix}^{\mathrm{T}} \tag{14}$$

The extended Kalman filter propagates the covariance of the state estimate at each time step using the Jacobian of the dynamic model evaluated at the current state. The nonzero elements of the Jacobian matrix A are given below

$$\frac{\partial \dot{\hat{p}}^i}{\partial \hat{v}^i} = I_{3\times 3}, \qquad \frac{\partial \dot{\hat{v}}^i}{\partial \hat{R}} = -\hat{L}_{ib}\widetilde{\hat{s}}, \qquad \frac{\partial \dot{\hat{v}}^i}{\partial \hat{s}_b} = -\hat{L}_{ib}, \tag{15}$$
$$\frac{\partial \dot{\hat{R}}}{\partial \hat{R}} = -\widetilde{\hat{\omega}}, \qquad \frac{\partial \dot{\hat{R}}}{\partial \hat{\omega}_b} = -I_{3\times 3},$$

where $\widetilde{a}$ indicates the skew symmetric matrix composed of the components of a.

The state covariance is propagated by calculating the covariance matrix derivative:

$$\dot{P}_{state} = AP_{state} + P_{state}A^{\mathrm{T}} + Q, \tag{16}$$

where $Q = \mathrm{diag}\left(0, Q_s, Q_\omega, Q_{s_b}, Q_{\omega_b}\right)$ is the process noise covariance and P_{state} is the 15×15 submatrix of the full covariance, P, corresponding to the state vector. Q_{s_b} and Q_{ω_b} are the process noise covariance for the acceleration and angular velocity bias states, respectively.

Database points are parameterized as Cartesian coordinates in the inertial frame.

$$\hat{p}_{fp} = \begin{bmatrix} \hat{p}_{fp_1} & \dots & \hat{p}_{fp_N} \end{bmatrix}^{\mathrm{T}} \in \mathbb{R}^{3N} \tag{17}$$

The covariance matrix for each feature point is P_{fp_j}. Database points are assumed to be static and have no process noise, therefore no propagation is necessary.

B. State and covariance update

The measurement model used in the navigation algorithm is the standard pinhole camera model. An undistortion transform is applied to the image features before being used in the algorithm, and so a simple camera model is appropriate.

Let the location of database point j be given by $r^c_{fp_j} = L_{ci}\left(p_{fp_j} - p\right) = \begin{bmatrix} X_j & Y_j & Z_j \end{bmatrix}^{\mathrm{T}}$, where L_{ci} is the transformation from the inertial to camera frame. The camera frame is defined such that the x axis is aligned with the optical axis, and the y and z axes point right and down on the image plane, respectively. Therefore, the camera model is given by

$$z_j = \begin{bmatrix} u \\ v \end{bmatrix}_j = g(x, p_{fp_j}) = \begin{bmatrix} f_u \frac{Y_j}{X_j} + \eta_u \\ f_v \frac{Z_j}{X_j} + \eta_v \end{bmatrix} \tag{18}$$

where it is assumed $\eta_u \sim N(0, R_{fp})$ and $\eta_v \sim N(0, R_{fp})$. Note $R = \mathrm{diag}\left(R_{fp}, R_{fp}\right)$.

The navigation algorithm performs the state update in a sequential manner. At each time step, a subset of the feature database is observed and used to update the state. For each feature j, the Jacobian of the measurement model is calculated, C^j_k.

$$C^j_k = \begin{bmatrix} \frac{\partial z_j}{\partial \hat{x}} & 0 & \dots & \frac{\partial z_j}{\partial \hat{p}_{fp_j}} & \dots & 0 \end{bmatrix} \tag{19}$$

As each feature is updated, C^j_k is reevaluated at the new best estimate. The state and covariance update is

calculated according to the familiar EKF update equations.

$$K_k = P_{k|k-1} C_k^{j\mathrm{T}} \left(C_k^j P_{k|k-1} C_k^{j\mathrm{T}} + R \right)^{-1} \tag{20}$$

$$\begin{bmatrix} \hat{x} \\ \hat{p}_{fp} \end{bmatrix}_{k|k} = \begin{bmatrix} \hat{x} \\ \hat{p}_{fp} \end{bmatrix}_{k|k-1} + K_k \left(z_j - g(\hat{x}_{k|k-1}, \hat{p}_{fp_j k|k-1}) \right) \tag{21}$$

$$P_{k|k} = \left(I - K_k C_k^{j\mathrm{T}} \right) P_{k|k-1} \tag{22}$$

A note on the notation: the "timestep" k is incremented on each feature update, even though no time has passed while updating measurements from a given frame, and therefore no propagation step has been carried out. In other words, for a given image, $\begin{bmatrix} \hat{x} \\ \hat{p}_{fp} \end{bmatrix}_{k+1|k} = \begin{bmatrix} \hat{x} \\ \hat{p}_{fp} \end{bmatrix}_{k|k}$ and $P_{state,k+1|k} = P_{state,k|k}$.

C. Point Matching

Before extracted features can be used in the EKF, it must first be determined which database point they correspond to. In previous work, a maximum-likelihood approach was used, where database points were corresponded to the measurement with the closest Mahalanobis distance.[7] In this paper we improve upon that method by using the Mahalanobis distance to narrow the pool of potential matches, and then using a descriptor to choose the best match. The use of a descriptor added robustness by limiting the number of false matches, especially in the presence of state error when the nearest statistical point may not be the best.

The feature detector is a modified Harris corner detector, which extracts points based on the image gradient in orthogonal directions.[14] To ensure good feature separation, the detector partitions the image into bins, and sets a maximum number of features in each bin. Also, a minimum distance between features can be set. The feature descriptor used to uniquely identify the point is the pixel intensity in a window surrounding the feature. The feature locations as well as the intensity descriptor is passed to the navigation algorithm in order of their Harris corner score, up to a maximum number.

The search region is determined using the estimated location uncertainty:

$$S^j = (C^j P C^{\mathrm{j}T} + R) \tag{23}$$

The statistical distance between measurement z_i and the image plane location of point p_{fp_j} is given by

$$e_{ij} \triangleq z_i - g(\hat{x}, \hat{p}_{fp_j}) \tag{24}$$

$$Z_{ij} = e_{ij}^{\mathrm{T}} (S^j)^{-1} e_{ij} \tag{25}$$

A maximum value, Z_{lim}, is set for Z_{ij} for measurement i and database point j to be considered as possible matches.

All measurements satisfying the Z_{ij} threshold are compared to the database point using their image descriptor. The average difference between pixel intensities is calculated and the pair with the lowest difference, below a maximum threshold, is considered to be corresponded to the database point.

The descriptor from the corresponded measurement is stored with the database point for use in the next iteration. This method, as opposed to retaining the original descriptor for the life of the database point, allows the appearance of the point to change slowly as the camera location changes, and makes it unnecessary to use more complex scale and rotation invariant descriptors.

If a measurement z_i does not meet the conditions to match any database point, it is considered to be a measurement of no currently stored point and is uncorresponded. Uncorresponded points can be used to initialize new database points, as described in the following section.

D. Initialization of new points

New points are initialized from uncorresponded features, and features with the highest corner score are initialized first. Because of the lack of depth information from a single monocular image, additional information is necessary to estimate the measurements' three-dimensional coordinates from their two-dimensional image plane location. In many common environments for vehicles with a camera aligned with the body z-axis, it is a reasonable assumption that the points lie on a plane at a distance of the altitude of the vehicle.This assumption would be valid, for instance, for an vehicle flying indoors over an uncluttered floor, or a outdoor vehicle flying over a plain. By initializing points with a suitable uncertainty in altitude, points that are in fact not on the ground plane will converge to the correct altitude if they remain in view.[12] Let the direction of the feature in the inertial frame be given by

$$h^i_{fp} = L_{ic} \begin{bmatrix} 1 \\ u/f_x \\ v/f_y \end{bmatrix} \tag{26}$$

Then the 3d location of the new feature is

$$h^i_{fp} \frac{X^i_j}{h^i_{fp_1}} = \begin{bmatrix} X_j \\ Y_j \\ Z_j \end{bmatrix}^i = r^i_{fp_j}. \tag{27}$$

$$\hat{p}_{fp_j} = r^i_{fp_j} + \hat{p} \tag{28}$$

Distance X^i_j is assumed to be the altitude of the camera, and image plane locations u and v and focal lengths f_x and f_y are known. The covariance of the new feature is set to fixed value scaled by the distance of the vehicle to that feature:

$$P_{fp_j} = L_{ic} \begin{bmatrix} ||r^i_{fp_j}||^2 & 0 & 0 \\ 0 & R_0||r^i_{fp_j}||^2/f_x^2 & 0 \\ 0 & 0 & R_0||r^i_{fp_j}||^2/f_y^2 \end{bmatrix} L_{ic}^{\mathrm{T}}. \tag{29}$$

where P_{fp_j} is the 3×3 block within the full covariance matrix P corresponding to the covariance of point p_{fp_j} with itself, and R_0 is a scalar initialization value. All other cross-correlation terms in P are set to zero.

Feature removal from the database is determined by comparing a "confidence index" of the current database features with the initialization confidence index of a new feature. The confidence index for current database features is limited to a minimum and maximum of 0 and 100, and is incremented for every iteration a feature is corresponded to a measurement, and decremented if it is not corresponded. The index can be thought of as a measure of how often a database feature is used (corresponded) for navigation in the recent past. The initialization confidence index is related to the number of database features which were corresponded on the current iteration. A database feature is replaced if its confidence index falls below the initialization confidence index. When few features are corresponded, the initialization confidence index is high, and features are replaced quickly. When many features are corresponded, the initialization confidence index is low and features are replaced slowly. The initialization confidence index represents a dynamic point removal threshold modified by the number of correspondences in the current iteration.

To demonstrate the effectiveness of the proposed method, it is compared to a fixed threshold feature removal method. The fixed threshold method is implemented by setting the initialization confidence index to 50 regardless of the number of features corresponded. This fixed threshold method is equivalent to the method proposed in 1 and 3, with the number of iterations equal to 50 and the percent correspondence equal to 50%.

The simulated vehicle was flown over a highly textured surface which was instantaneously changed to a low textured surface. This is similar to the effect of instantaneous lighting changes, such as a light being switched off in a room. The texture removal was performed 8 times for each method, and the average, minimum and maximum number of feature correspondences is shown in Figure 3. It can be seen that the dynamic threshold does a better job maintaining a steady number of consistent features throughout the test. When the texture is removed, the fixed threshold takes a longer time to throw out the now-unobservable features, as the confidence index for these points fall. In the dynamic thresholding case, the threshold is raised when few features correspond, and so unseen features are removed quickly. The dynamic threshold method allows the navigation algorithm to rapidly adapt to dynamic environment.

III. Parameter Selection Study

The accuracy and precision of the vision-based navigation system presented in this paper is affected by a variety of factors. These include the image texture, video noise level, distance from mapped points, and aggressiveness of flight. These factors differ significantly from those that influence GPS, and thus it is necessary to examine the influence these conditions have on vehicle performance. The primary contribution of this work is an examination of the vehicle performance as flight conditions change and the identification of navigation and controller parameters which can be used to improve performance over the range of conditions. Specifically, the relationship between the distance of the vehicle from mapped features, essentially the altitude of the vehicle, and vehicle performance are investigated. The metrics used to evaluate the system performance are navigation accuracy during hover and rejection of navigation error.

A. Motivating example

A simple example will help motivate the results by illustrating the effect of altitude on the navigation solution: Consider a two-state vehicle model, and a measurement model consisting of the direction of the feature point from the vehicle. The state vector is given by position and velocity along the x-axis, $[x, v]$, and height h is held constant. See Figure 4 for schematic of the problem.

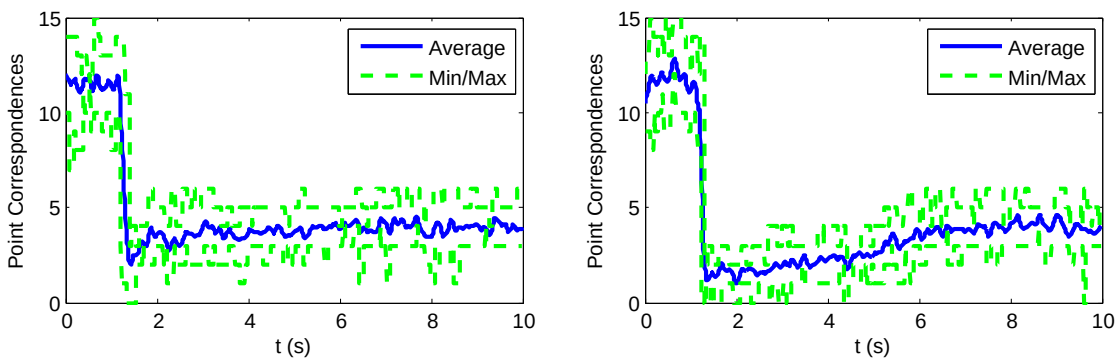


Figure 3. The number of feature correspondences over time. The left plot data was gathered using the dynamic thresholding used in this paper. The right plot uses a fixed threshold and is shown for comparison. After the change in image texture at $t = 1$, the dynamic threshold method more quickly removes old points from the database.

Consider two possible measurement models

$$y_1 = x \tag{30}$$

$$y_2 = \phi = \operatorname{atan}\left(\frac{x}{h}\right) \tag{31}$$

Measurement model y_1 is the result of directly measuring the x location of the vehicle, similar to GPS measurement. Measurement model y_2 is the result of measuring the bearing to a reference point, as would be the case when using a camera. Linearizing results in a measurement matrices given by

$$C_1 = \begin{bmatrix} 1, & 0 \end{bmatrix} \tag{32}$$

$$C_2 = \begin{bmatrix} \frac{1}{h}, & 0 \end{bmatrix} \tag{33}$$

Now consider the algebraic Riccati equation, the solution of which, P gives the uncertainty of the position and velocity estimate.

$$0 = PA^{\mathrm{T}} + AP + Q - PC^{\mathrm{T}}(CPC^{\mathrm{T}} + R)^{-1}CP \tag{34}$$

where A is the dynamic model, Q is the process noise and R is the measurement noise.

The measurement model affects the final term in the equation. Inserting and performing basic algebraic manipulation we note that

$$PC_2^{\mathrm{T}}(C_2PC_2^{\mathrm{T}} + R)^{-1}C_2P = \frac{1}{h^2}\begin{bmatrix} p_1^2 & p_{12}p_1 \\ p_{12}p_1 & p_{12}^2 \end{bmatrix}\frac{1}{\frac{p_1}{h^2} + R} \tag{35}$$

$$= \begin{bmatrix} p_1^2 & p_{12}p_1 \\ p_{12}p_1 & p_{12}^2 \end{bmatrix}\frac{1}{p_1 + Rh^2} \tag{36}$$

$$= PC_1^{\mathrm{T}}(C_1PC_1^{\mathrm{T}} + Rh^2)^{-1}C_1P \tag{37}$$

where $P = \begin{bmatrix} p_1 & p_{12} \\ p_{12} & p_2 \end{bmatrix}$. This shows that using a bearing only sensor such as a camera is equivalent to directly measuring the position with increasing uncertainty as altitude increases. This example shows that altitude has a profound effect on the performance of the navigation solution.

B. Simulation results

The evaluation of the system was conducted using the Georgia Tech UAV Simulation Toolbox (GUST). The GUST software package that combines a high-fidelity vehicle and environment model, onboard flight control

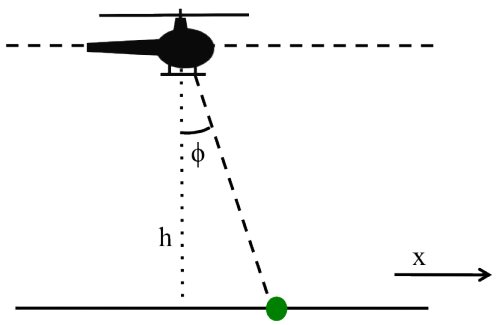


Figure 4. Schematic of a 2-D helicopter model using vision-based navigation.

software, and ground station software. GUST may be operated in hardware in the loop (HITL) mode or software in the loop (SITL) mode. In HITL mode, the flight control software and ground station interface with physical sensors, actuators, and communication links. In SITL mode, the flight control software and ground station interface with the vehicle model and simulated communication links. This design ensures that the same software is used in simulation and in flight. The vehicle model is a six rigid body degree of freedom model and simulates sensor noise, delay, location, orientation, and actuator dynamics and saturation. The vehicle model can also simulate external disturbances such as turbulence and wind.

The navigation parameters examined were the measurement covariance, R, and the statistical matching region limit, Z_{lim}, described in section II.D. The measurement covariance is the amount of uncertainty in the location of the features extracted from the images. The feature locations are normalized by the screen size, where the upper left corner is $(-1,-1)$ and the lower right is $(1,1)$, and therefore $1/4$ pixel^2 covariance is approximately 1.3×10^{-5} for a camera resolution of 320×240. The statistical matching region limit is the limit on the statistical distance a feature can be from the database point and still be considered as a possible match. Z_{lim} is measured in square of the standard deviation from the predicted point location. The size of the search ellipse is determined in part by the trade-off between R and Z_{lim}: increasing R tends to increase the matching region, while decreasing Z_{lim} will decrease the matching region.

During the simulation study, R was increased and Z_{lim} was decreased, which increased the uncertainty of the measurement while keeping the matching radius a reasonable size. Two tests were conducted at each pair of R and Z_{lim} values: a 90 second hover, and 2 and 3 ft/s velocity error injections into the state vector. For the hover test, the error between the navigation solution and the true position was used as the performance metric. For the error injection test, the velocity error was injected five times for each case, and the average time for the navigation error to settle to the true velocity and the average final absolute position error were recorded. These tests were performed at three altitudes, 4 ft, 16 ft and 64 ft. The image on the ground was enlarged proportionally, so that the feature quality was consistent across all series.

The first series of tests was conducted at an altitude of 4 ft. Figure 5 show the results. In Figure IV, the hover test results are shown with the values for R and Z_{lim} given. R and Z_{lim} are started at $1/4$ pixel^2 uncertainty and $\sqrt{.5}\sigma$ uncertainty limit, respectively. As the parameters are changed, the size of the navigation error gets smaller. It would seem at first glance that the increasing R and reducing Z_{lim} is an unmitigated good. However, the Figure 5(b) shows that a trade-off is being made. The case number along the x-axis each subfigure corresponds to the parameter pair in Figure IV. It can be seen that as the case number increases, the settling time increases exponentially and the final error is increased. The error injection results show that the system is less robust to velocity error as R is decreased and Z_{lim} is increased. Case 4 ($R=.013, Z_{lim}=.5$) has been chosen as the best compromise between robustness and performance at this altitude.

A second test was performed at a higher altitude, 16ft, and the results are shown in Figure 6. It can be seen in Figure 6(a) that increasing the distance to the mapped points causes the precision of the navigation solution to decrease. Furthermore, Figure 6(b) shows that error rejection settling time is increased for all cases indicating a decrease in robustness. Case 4 ($R=.013, Z_{lim}=.5$) again provides the best compromise, but the acceptable region for these settings have narrowed, especially in the direction of reduced R. In fact, several case results have been omitted because they caused catastrophic navigation failure and vehicle crash.

A third test was performed at a higher altitude, 64 ft, and the results are shown in Figure 7. Hover accuracy was decreased further, as would be expected from the fact that a translation at 64 ft altitude causes very little change in the location of points in the image. At high-R, the increase in accuracy during hover reaches a limit and the vehicle position begins to drift. Data is omitted from cases which caused vehicle crashes. As can be seen in Figure 7(b), the settling time and final position error are increased. Case 5 ($R=.13, Z_{lim}=.1$) is the best compromise at this altitude, and only cases 5 and 6 provide stable navigation solutions in the error injection tests.

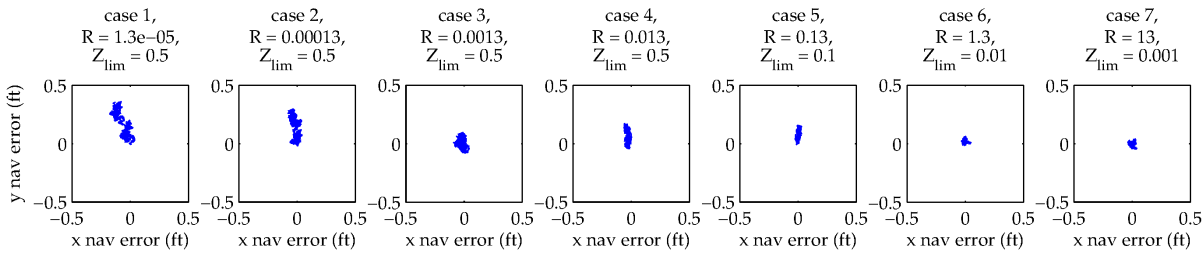
This study shows that vision-aided inertial navigation can exhibit a large performance variation, to the

point of becoming unflyable, in varying flight conditions.

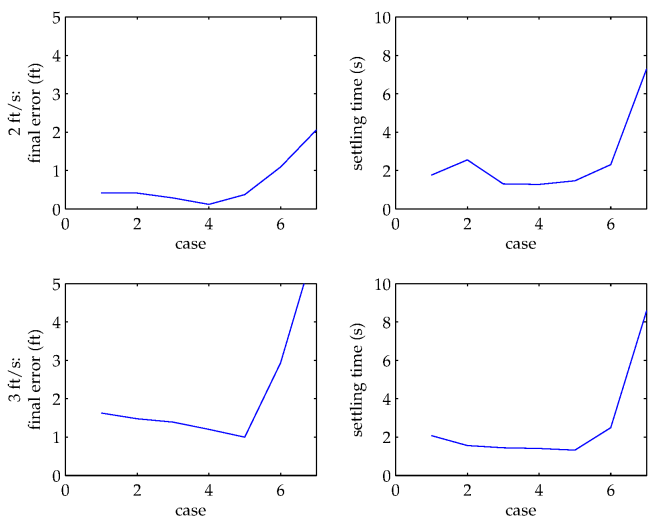
IV. Flight Test Results

The vision-aided inertial navigation system presented in this paper was flight tested on a quadrotor miniature UAV, the GTQ3. The GTQ3 weighs 1.5 kg and has an overall diameter of 80 cm. A downward-facing camera mounted on the vehicle provides 320x240 resolution images of the ground which are processed in the navigation system at approximately 9 fps. The GTQ3 is controlled by a model-inversion adaptive controller with pseudocontrol hedging, described in detail in Ref. 15. The entire navigation solution is processed onboard the vehicle on a 1.4Ghz quad-core ARM-based computer.

Figure 8 shows the navigation solution in a hover at an altitude of 4 ft. Each plot shows data gathered at navigation parameters roughly corresponding to a case from the simulation test shown in Figure 5. The plots are of the navigation solution, in contrast to the navigation error given in the previous plots, because there was no ground truth available to compare to as in the simulation. The data demonstrates that a stable navigation solution was possible in all cases, including case 4, which was predicted to be the best at this altitude.



(a)



(b)

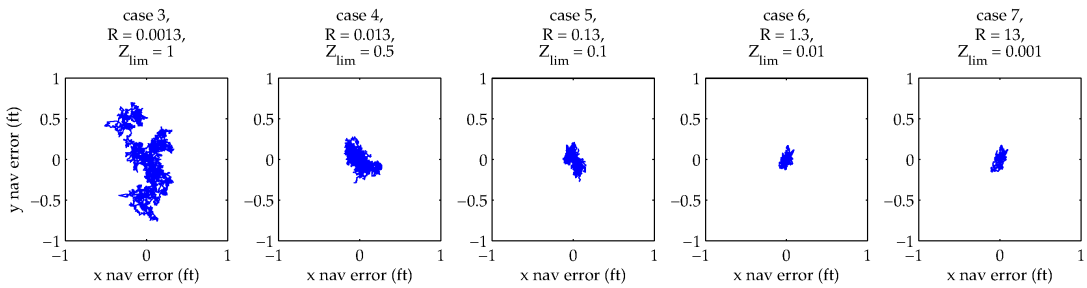
Figure 5. Plots illustrating the performance of the navigation solution during hover and in response to error injection at 4 ft altitude for various settings of R and Z_{lim}. It can be seen that the accuracy of the navigation solution during hover improves as R is decreased, but the settling time after error injection is increased.

V. Conclusion

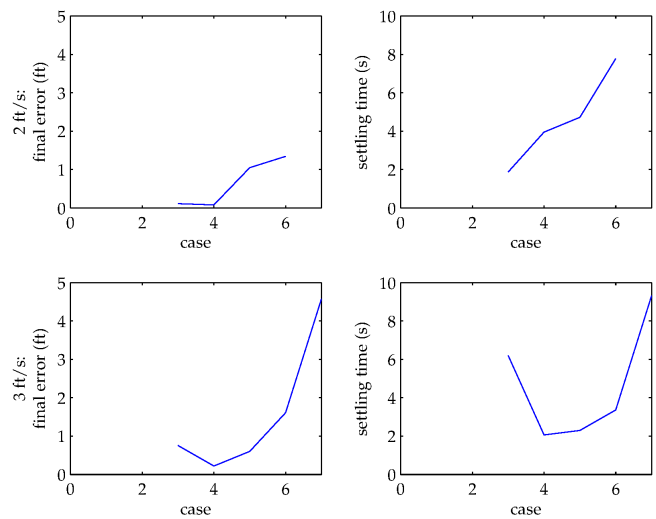
This paper presents an analysis of the performance of an vision-aided inertial navigation system over large altitude range. As expected, the navigation precision deteriorates as the altitude increases, due to the distance of the database points from the vehicle. As altitude increases, translation of the vehicle causes progressively smaller motion of the features in the image plane. This loss of precision renders low-altitude navigation and controller parameters inadequate or even unflyable. Flight test results are presented which verify the conclusions of the simulation study.

Acknowledgments

Many thanks to Dmitry Bershadsky and Stephen Haviland for help with flight testing. This work was performed under the NIST grant no. $70NANB10H013$.

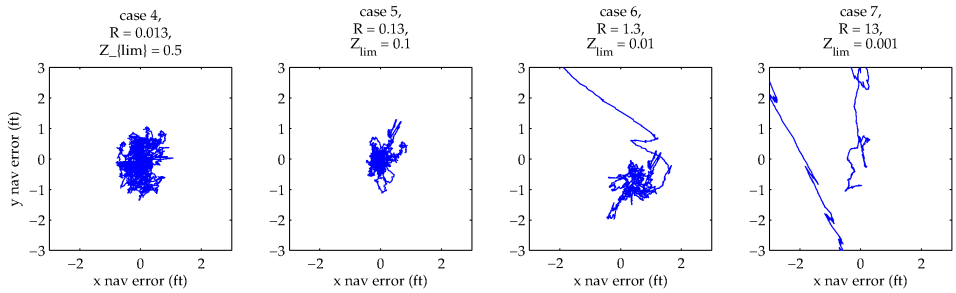


(a)

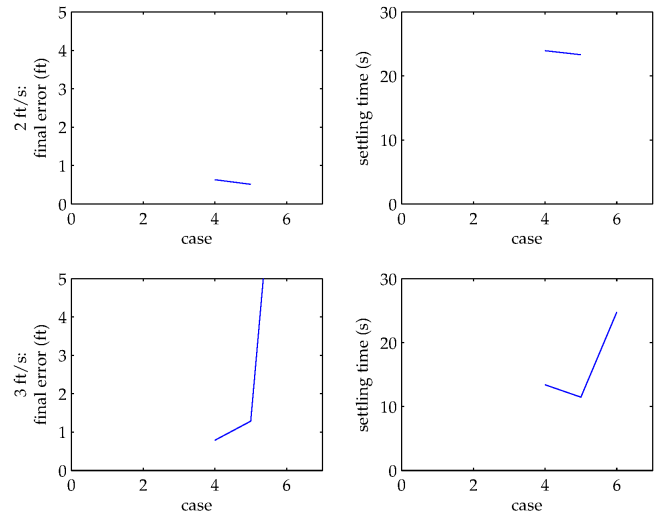


(b)

Figure 6. Plots illustrating the performance of the navigation solution during hover and in response to error injection at 16 ft altitude for various settings of R and Z_{lim}. The precision of the navigation solution is decreased with altitude as expected, and settling times have increased.



(a)



(b)

Figure 7. Plots illustrating the performance of the navigation solution during hover and in response to error injection at 64 ft altitude for various settings of R and Z_{lim}. The precision of the navigation solution is decreased still further, to the point that the vehicle is unflyable in several cases.

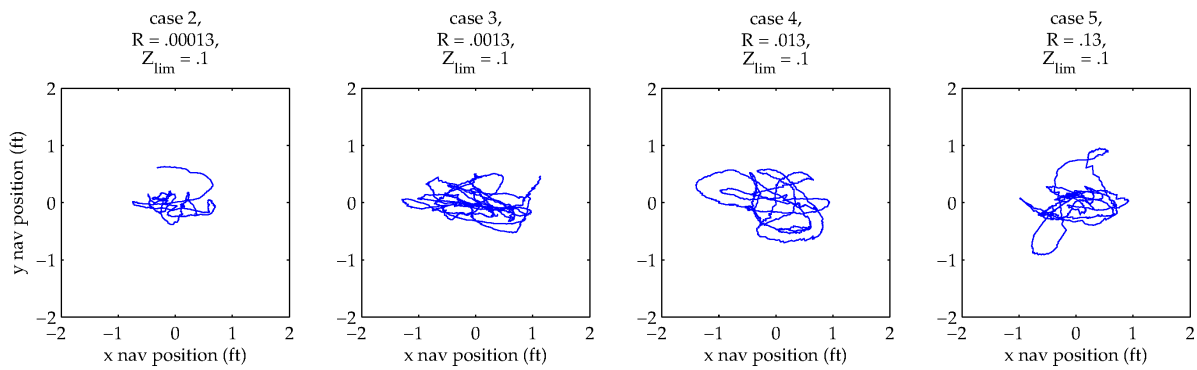


Figure 8. Flight test results illustrating the performance of the navigation solution during hover at 4 ft altitude for various settings of R and Z_{lim}. All four cases demonstrate a stable navigation solution at this altitude.

References

[1]A. Davison, "Real-time simultaneous localisation and mapping with a single camera," in *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, oct. 2003, pp. 1403 –1410 vol.2.

[2]G. Klein and D. Murray, "Parallel tracking and mapping for small ar workspaces," in *Mixed and Augmented Reality, 2007. ISMAR 2007. 6th IEEE and ACM International Symposium on*, nov. 2007, pp. 225 –234.

[3]A. Davison, I. Reid, N. Molton, and O. Stasse, "Monoslam: Real-time single camera slam," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 29, no. 6, pp. 1052 –1067, june 2007.

[4]E. Eade and T. Drummond, "Scalable monocular slam," in *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, vol. 1, june 2006, pp. 469 – 476.

[5]J. Civera, A. Davison, and J. Montiel, "Inverse depth parametrization for monocular slam," *Robotics, IEEE Transactions on*, vol. 24, no. 5, pp. 932 –945, oct. 2008.

[6]G. Sibley, C. Mei, I. Reid, and P. Newman, "Vast-scale outdoor navigation using adaptive relative bundle adjustment," *The International Journal of Robotics Research*, vol. 29, no. 8, pp. 958–980, 2010. [Online]. Available: http://ijr.sagepub.com/content/29/8/958.abstract

[7]G. Chawdhary, E. N. Johnson, D. Magree, A. Wu, and A. Shein, "Gps-denied indoor and outdoor monocular vision aided navigation and control of unmanned aircraft," *Journal of Field Robotics*, Accepted, 2013.

[8]S. Weiss, D. Scaramuzza, and R. Siegwart, "Monocular-slambased navigation for autonomous micro helicopters in gps-denied environments," *Journal of Field Robotics*, vol. 28, no. 6, pp. 854–874, 2011. [Online]. Available: http://dx.doi.org/10.1002/rob.20412

[9]M. Achtelik, M. Achtelik, S. Weiss, and R. Siegwart, "Onboard imu and monocular vision based control for mavs in unknown in- and outdoor environments," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, may 2011, pp. 3056 –3063.

[10]S. Weiss, M. Achtelik, S. Lynen, M. Chli, and R. Siegwart, "Real-time onboard visual-inertial state estimation and self-calibration of mavs in unknown environments," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, may 2012, pp. 957 –964.

[11]S. Scherer, J. Rehder, S. Achar, H. Cover, A. Chambers, S. Nuske, and S. Singh, "River mapping from a flying robot: state estimation, river detection, and obstacle mapping," *Autonomous Robots*, vol. 33, pp. 189–214, 2012. [Online]. Available: http://dx.doi.org/10.1007/s10514-012-9293-0

[12]A. Wu and E. N. Johnson, "Autonomous flight in gps-denied environments using monocular vision and inertial sensors," in *Infotech@AIAA*. Atlanta, GA: AIAA, April 2010.

[13]A. D. Wu, E. N. Johnson, M. Kaess, F. Dellaert, and G. Chowdhary, "Autonomous flight in gps-denied environments using monocular vision and inertial sensors," *AIAA journal of aerospace computing, information, and communication*, 2012, accepted.

[14]C. Harris and M. Stephens, *A combined corner and edge detector*. Manchester, UK, 1988, vol. 15, no. Manchester, pp. 147–151. [Online]. Available: http://www.cis.rit.edu/ cnspci/references/dip/harris1988.pdf

[15]E. N. Johnson and S. K. Kannan, "Adaptive trajectory control for autonomous helicopters," *AIAA Guid. Contr. Dyn.*, vol. 28, pp. 524–538, 2005.